IN THE UNITED STATES PATENT OFFICE

Applicant:	Reto Stamm et al.		
Assignee:	Xilinx, Inc.		
Title:	SYSTEM AND METHOD FOR TESTING PARAMETERIZED LOGIC CORES		
Ser. No.:	09/360,472	Filing Date:	07/23/1999
Examiner:	Thai Q. Phan	Art Unit:	2128
Docket No.:	X-528 US	Conf. No.:	4229

Mail Stop Appeal Brief - Patents
Board of Patent Appeals and Interferences
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

Sir:

This Appeal Brief is submitted pursuant to 37 C.F.R. §41.37, in support of the Notice of Appeal filed November 23, 2004.

I. Real Party In Interest

The real party in interest is Xilinx, Inc., having a place of business at 2100 Logic Drive, San Jose, California 95124-3400. The above referenced patent application is assigned to Xilinx, Inc.

II. Related Appeals and Interferences

Appellant is unaware of any related appeals, interferences or judicial proceedings.

III. Status of Claims

Claims 1-20 are pending and are presented for appeal. Claims 1-7 and 15-20 stand rejected under 35 U.S.C. §103(a) as being obvious over US patent no. 6,120,549 to Goslin in view of US patent no. 6,553,531 to Kim.

Claims 8-14 would be allowable if rewritten to overcome the rejection(s) under 35 U.S.C. §112, second paragraph, and to include the limitations of the base claim and any intervening claims. As to the alleged deficiency relative to 35 U.S.C. §112, second paragraph, no specific failings have been cited, and the claims are thought to comply with the requirements of 35 U.S.C. §112, second paragraph.

The claims as currently pending may be found in the attached Appendix of Appealed Claims.

IV. Status of Amendments

No amendments have been made to the claims.

V. Summary of the Invention

One embodiment of Appellant's invention is directed to a method for testing a parameterizable logic core. A set of parameter values for the logic core is randomly generated (FIG. 1, 102; p. 4, l. 5 – p. 5, l. 22; FIG. 4; p. 9, l. 7; FIG. 5; p. 10, l. 1), and a netlist is generated from the parameter values (FIG. 1, 106; p. 5, l. 23). Circuit behavior is then simulated to test the logic core (FIG. 1, 108; p. 5, l. 30).

In another embodiment, a system is provided for testing a parameterizable logic core. The system includes a test controller configured and arranged to randomly generate a set of parameter values for the logic core (FIG. 2, 162; FIG. 3, 186; p. 7, l. 32; p. 8, l. 23). A core generator is coupled to the test controller and is configured and arranged to generate a netlist from the logic core and set of parameter values (FIG. 2, 164; FIG. 3, 182). A simulator is coupled to the test controller and is configured and arranged to simulate circuit behavior with the netlist and a predetermined test bench (FIG. 2, 166).

In yet another embodiment, a GUI-based system is provided for testing a parameterizable logic core. The system includes a test controller including a GUI-driver. The GUI-driver is configured and arranged to randomly generate a set of parameter values for the logic core (FIG. 3, 186; p. 8, l. 23). A core generator includes a GUI coupled to the GUI-driver. The core generator is configured and arranged to generate a netlist from the logic core and a set of parameter values (FIG. 3, 182) A simulator is coupled to the test controller, and the simulator is configured and arranged to simulate circuit behavior with the netlist and a predetermined test bench (FIG. 3, 166).

An apparatus for testing a parameterizable logic core is provided in another embodiment. The apparatus includes means for randomly generating a set of parameter values for the logic core (FIG. 1, 102; p. 4, l. 5 – p. 5, l. 22; FIG. 4; p. 9, l. 7; FIG. 5; p. 10, l. 1; FIG. 2, 162; FIG. 3, 186; p. 7, l. 32; p. 8, l. 23); means for generating a netlist from the set of parameter values and logic core(FIG. 1, 102; p. 4, l. 5 – p. 5, l. 22; FIG. 4; p. 9, l. 7; FIG. 5; p. 10, l. 1; FIG. 2, 164; FIG. 3, 182); and means for simulating circuit behavior with the netlist (FIG. 1, 108; p. 5, l. 30; FIG. 2, 166).

VI. Grounds of Rejection

Claims 1-7 and 15-20 stand rejected under 35 U.S.C. §103(a) over "Goslin" (U.S. Patent No. 6,120,549), in view of "Kim" (U.S. Patent No. 6,553,531).

VII. Argument

The rejection of claims 1-7 and 15-20 is improper because the Examiner fails to establish a *prima facie* case of obviousness.

Claims 1, 15, 18, 19, and 20

The Examiner fails to establish that all the limitations of the claims are suggested by the Goslin-Kim combination, fails to provide a proper motivation for combining teachings of Kim with Goslin, and fails to and show that the combination could be made with a reasonable likelihood of success.

The claims include limitations of and related to randomly generating a set of parameter values for the logic core. A netlist is generated from the parameter values, and circuit behavior is then simulated to test the logic core. None of the cited prior art that suggests randomly generating parameter values and generating a netlist from the random parameter values.

Goslin is cited as teaching specifying values for parameters of a system-level module and generating a netlist file from the module. The teachings of Kim do not suggest that values for these parameters may be randomly generated, and with those random parameters a netlist may be built. Kim suggests randomly generating stimuli during simulation. Thus, even if teachings of Kim were combined with teachings of Goslin, the resulting system would not involve randomly generating parameter values for a logic core and generating a netlist using those random parameter values.

Kim's teachings specifically indicate that the random data is generated for input to simulation, not random parameter values for generating a netlist from a logic core. For example, Kim's title calls out random stimulus generation. Furthermore, Kim teaches:

> The Verilog simulator executes a Verilog language model of a hardware device under test, while the Vera simulator executes a Vera language model of the environment in which the DUT is to be tested. As an environment simulator, the basic functionality of the Vera simulator is to simulate the DUT by driving certain of its inputs and to monitor the resulting states of the DUT by sampling the values of its nodes. (col. 6, ll. 48-55).

Kim's invention adds to the Vera simulator capabilities which facilitate the generation of random data. (col. 10, ll. 30-32). Kim's random data is for input stimuli to the simulator, not random parameter values for a logic core from which a netlist is generated. Furthermore, Kim does not suggest the capability to generate random data for simulation stimuli may be extended to parameter values. Kim's disclosure appears to be limited to simulation and does not appear to discuss the process of generating a netlist. Thus, Kim's teachings are not suggestive of the present claims.

The alleged motivation for modifying Goslin with Kim does not support *prima facie* obviousness. The alleged motivation states, "This would motivate practitioner in the art at the time of the invention was made to combine Kim teaching of random value generation of design variables in the circuit design test and verification such as in Goslin parameterized logic cores in order to carry out faster and accurate test and verification of the design as taught in Kim." This alleged motivation is improper because it is conclusory.

Addressing the "rigorous ... requirement for a showing of the teaching or motivation to combine prior art references," the Court of Appeals for the Federal Circuit has stated:

> We have noted that evidence of a suggestion, teaching, or motivation to combine may flow from the prior art references themselves, the knowledge of one of ordinary skill in the art, or, in some cases, from the nature of the problem to be solved, (citations omitted), although "the suggestion more often comes from the teachings of the pertinent references," *Rouffet*, 149 F.3d at 1355, 47 USPQ2d at 1456. The range of sources available, however, does not diminish the requirement for actual evidence. That is, the showing must be clear and particular. *See, e.g., C.R. Bard*, 157 F.3d at 1352, 48 USPQ2d at 1232. Broad conclusory statements regarding the teaching of multiple references, standing alone, are not "evidence." *In re Dembiczak*, 175 F.3d 994, 50 U.S.P.Q.2d 1614 (Fed. Cir. 1999).

To the extent that the alleged motivation is understood, alleged motivation seems to say that the combination would provide faster and more accurate test and verification of the design. However, no evidence is provided that the testing and verification would be faster and more accurate, nor is there any reasoning provided as to how the combination would be faster and more accurate, nor is there any explanation of what is meant by "faster" and "more accurate". The alleged motivation is merely a broad conclusory statement, and no evidence has been provided that suggests how Goslin would or could be modified, nor has evidence been provided that suggests Goslin would benefit from such modification. Therefore, the alleged motivation is insufficient to support *prima facie* obviousness.

Successfully modifying Goslin with teachings of Kim is not reasonably likely. Goslin deals with generating optimized functional macros (Title), and Kim teaches generating random stimuli for simulation. Using random stimuli in the process of generating functional macros would not serve any apparent useful purpose because there is no simulation involved. Therefore, the modification would not be expected to reasonably succeed.

The Office Action does not establish a *prima facie* case of obviousness for claims 1, 18, 19, and 20 because the Office Action has not shown that all the limitations are taught by the references, has not provided a proper motivation to combine the references, and has not shown that Goslin could be successfully modified with the teachings of Kim with reasonable success.

Claim 15 depends from claim 1 and is patentable over the Goslin-Kim combination for at least the reasons set forth above.

Claim 2

Claim 2 includes limitations of and related to generating a random parameter value within predetermined upper and lower limits, and generating a new random parameter value if the random parameter value fails to meet predetermined criteria. The cited sections of Goslin teach displaying value ranges for parameters of a selected module (col. 6, ll. 51-62) and a module broker determining whether a suitable module exists based on user-input specific parameter values (col. 7, l. 52 – col. 8, l. 10). These teachings of Goslin appear in no discernable way to suggest the claim limitations. Therefore, the Examiner fails to show that the limitations of claim 2 are suggested by the Goslin-Kim combination.

Claim 3

Claim 3 includes limitations of and related to assigning respective probabilities to numbers between upper and lower limits for the parameters and generating the random parameter value as a function of the probabilities. The Examiner alleges that these limitations are suggested by Goslin at col.9, l. 20 – col. 10, l. 14. However, this section of Goslin suggests attaching weights to parameters for purposes of a module broker selecting between multiple modules that may satisfy user-input parameter values (col. 9, ll. 4-19). There is no apparent indication that Goslin generates random parameter values between the upper and lower limits as claimed. Thus, the Examiner fails to show that the limitations of claim 3 are suggested by the Goslin-Kim combination.

Claim 4

Claim 4 includes limitations of and related to providing a parameter value as input to a GUI and generating random replacement values for invalid values detected by the GUI. The Examine cites various sections of Goslin as suggesting these limitations. However, these sections merely indicate that a user may enter parameters and select modules via a GUI. There is no apparent suggestion of providing the randomly generated parameter values as input to a GUI and generating replacement values when the GUI detects invalid values. Thus, the limitations of claim 4 have not been shown to be suggested by the Goslin-Kim combination.

Claim 5

Claim 5 includes limitations of and related to providing the randomly generated set of parameter values to a GUI and identifying invalid parameter values with the GUI. Goslin's cols. 6-8 were cited as suggesting these limitations. However, there is no apparent suggestion of inputting the randomly generated parameter values to the GUI, nor of identifying invalid values with the GUI. Claim 5 is therefore not shown to be suggested by the Goslin-Kim combination.

Claim 6

Claim 6 depends from claim 5 and includes limitations of and related to generating replacement parameters values for invalid values. The rejection cites Goslin's col. 8, ll. 15-60) as suggesting these limitations. However, this section appears to discuss representation of spaces of possible parameter values (col. 8, l. 15) and choosing a module according to an area of a parameter space of the module (col. 8, ll. 40-46). There is no apparent suggestion of generating replacement parameter values. Therefore, the limitations of claim 6 are not shown to be suggested by the Goslin-Kim combination.

Claims 7 and 16

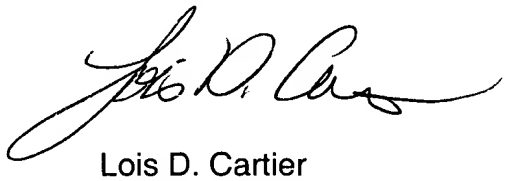
Claims 7 and 16 include limitations of and related to selecting a random order in which to provide parameter values to the graphical user interface and providing the parameters one-by-one as input to the graphical user interface. The rejection cites Goslin's col. 8, ll. 15-60) as suggesting these limitations. However, as is clear from the explanation of this section given above in reference to claim 6, this section of Goslin appears to have no relationship to selecting a random order in which to provide parameter values to the graphical user interface. The rejection of claims 7 and 16 fails because the Examiner has not shown that the Goslin-Kim combination suggests these limitations.

VIII. Conclusion

This is the second appeal of a final rejection of the claims in this application, and the claims of this application have not changed from the original application filing. Based on the Appeal Brief filed January 30, 2003, prosecution was reopened and new prior art references were applied in rejecting the claims. As explained above, the presently cited prior art does not suggest the claimed invention, and the latest round of Office Actions have seemingly done little in the way of advancing prosecution. Applicants respectfully request that, before any further Office Actions are issued, careful consideration be given to the teachings of any newly found prior art resulting from a follow-on search.

In view of the above, Appellant respectfully submits that the claimed invention is patentable over the cited prior art, and that the rejections of Claims 1-20 are in error. Appellant respectfully requests reversal of the rejections as applied to the appealed claims and allowance of the application.

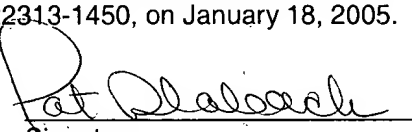
Respectfully submitted,



Lois D. Cartier
Agent for Applicants
Reg. No.: 40,941
(720) 652-3733

I hereby certify that this correspondence is being deposited with the United States Postal Service as first-class mail in an envelope addressed to: Commissioner for Patent, P.O. Box 1450 Alexandria, VA 22313-1450, on January 18, 2005.

Pat Slaback
Name



Signature

APPENDIX OF APPEALED CLAIMS

1. (Original) A computer-implemented method for testing a parameterizable logic core, comprising:

randomly generating a set of parameter values for the logic core;

generating a netlist from the set of parameter values and logic core;

simulating circuit behavior with the netlist.

2. (Original) The method of claim 1, further comprising for each parameter:

generating a random parameter value within predetermined upper and lower limits associated with the parameter; and

generating a new random parameter value if the random parameter value fails to meet predetermined criteria.

3. (Original) The method of claim 2, further comprising:

assigning respective probabilities to one or more numbers between the upper and lower limits for one or more of the parameters; and

generating the random parameter value as a function of the probabilities.

4. (Original) The method of claim 3, further comprising:

providing the parameter value as input to a graphical user interface;

generating random replacement values for invalid parameter values detected by the graphical user interface.

5. (Original) The method of claim 1, further comprising:

providing the set of parameter values to a graphical user interface; and

identifying invalid parameter values with the graphical user interface.

6. (Original) The method of claim 5, further comprising:

generating random replacement parameter values for the invalid parameters; and

repeating the steps of providing the replacement values to the graphical user interface, identifying invalid parameter values, and generating random replacements until all the parameter values are valid.

7. (Original) The method of claim 5, further comprising:

selecting a random order in which to provide parameter values to the graphical user interface; and

providing the parameters one-by-one as input to the graphical user interface.

8. (Original) The method of claim 5, further comprising:

cloning the set of parameter values;

mutating the set of parameter values, whereby a mutated set of parameter values is produced;

generating a new netlist from the mutated set of parameter values and logic core; and

simulating circuit behavior with the new netlist.

9. (Original) The method of claim 8, wherein the set of parameter values is cloned and mutated only if an error is detected in simulating the circuit behavior.

10. (Original) The method of claim 1, further comprising:

cloning the set of parameter values;

mutating the set of parameter values, whereby a mutated set of parameter values is produced;

generating a new netlist from the mutated set of parameter values and logic core; and

simulating circuit behavior with the new netlist.

11. (Original) The method of claim 10, wherein the set of parameter values is cloned and mutated only if an error is detected in simulating the circuit behavior.

12. (Original) The method of claim 11, further comprising repetitively cloning and mutating sets of parameters when errors are detected in simulating the circuit behavior, whereby multiple generations of sets of parameters are created.

13. (Original) The method of claim 12, wherein one or more of the parameter values in a parameter set are mutated.

14. (Original) The method of claim 13, wherein a number of parameter values mutated in a set of parameters is a function of a number of generations previously created.

15. (Original) The method of claim 1, further comprising:

identifying which parameters of a graphical user interface are editable; and

providing the set of parameter values to the graphical user interface.

16. (Original) The method of claim 15, further comprising:

selecting a random order in which to provide the parameter values to the graphical user interface; and

providing the values one-by-one as input to the graphical user interface.

17. (Original) The method of claim 1, further comprising:

accumulating respective numbers of tests having been performed using different parameter values

accumulating respective numbers of tests failed using each of the parameter values; and

highlighting parameters having numbers of failed tests equal to the number of tests.

18. (Original) A system for testing a parameterizable logic core, comprising:

a test controller configured and arranged to randomly generate a set of parameter values for the logic core;

a core generator coupled to the test controller, the core generator configured and arranged to generate a netlist from the logic core and set of parameter values;

a simulator coupled to the test controller, the simulator configured and arranged to simulating circuit behavior with the netlist and a predetermined test bench.

19. (Original) A system for testing a parameterizable logic core, comprising:

a test controller including a GUI-driver, the GUI-driver configured and arranged to randomly generate a set of parameter values for the logic core;

a core generator including a GUI coupled to the GUI-driver, the core generator configured and arranged to generate a netlist from the logic core and set of parameter values;

a simulator coupled to the test controller, the simulator configured and arranged to simulating circuit behavior with the netlist and a predetermined test bench.

20. (Original) An apparatus for testing a parameterizable logic core, comprising:

means for randomly generating a set of parameter values for the logic core;

means for generating a netlist from the set of parameter values and logic core; and

means for simulating circuit behavior with the netlist.

APPENDIX OF EVIDENCE

Appellant is unaware of any evidence submitted in this application pursuant to 37 C.F.R. §§ 1.130, 1.131, and 1.132.

APPENDIX OF RELATED PROCEEDINGS

As stated in Section II above, Appellant is unaware of any related appeals, interferences or judicial proceedings.